(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 777 178 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
04.06.1997 Bulletin 1997/23

(51) Int. Cl.[6]: **G06F 9/44**

(21) Application number: **96308493.4**

(22) Date of filing: **25.11.1996**

(84) Designated Contracting States:
DE FR GB

(30) Priority: **30.11.1995 US 565374**

(71) Applicant: **INTERNATIONAL BUSINESS MACHINES CORPORATION**
Armonk, NY 10504 (US)

(72) Inventor: **Coskun, Nurcan**
Austin, Texas 78727 (US)

(74) Representative: **Waldner, Philip**
IBM United Kingdom Limited,
Intellectual Property Department,
Hursley Park
Winchester, Hampshire SO21 2JN (GB)

(54) Data processing system

(57) Providing dynamic roles for objects in an object-oriented programming environment. A mechanism adds roles dynamically for an object depending on the context of the object. The IBM System Object Model (SOM) is the principal building block for providing this function. The dispatch resolution of the SOM is used to provide the functions of this invention the "somDispatch" methods feature. The "somDispatch" methods is overridden in new classes to implement application specific dispatch mechanism. Dynamic objects are represented as lists of objects where objects are added to the list at any time. For example, when a program is started, a person object will only have a student object. Before calling a function requiring teacher characteristics, the teacher role is added to the person object. When the function call returns, the teacher role is deleted from the person object if the role is no longer needed.

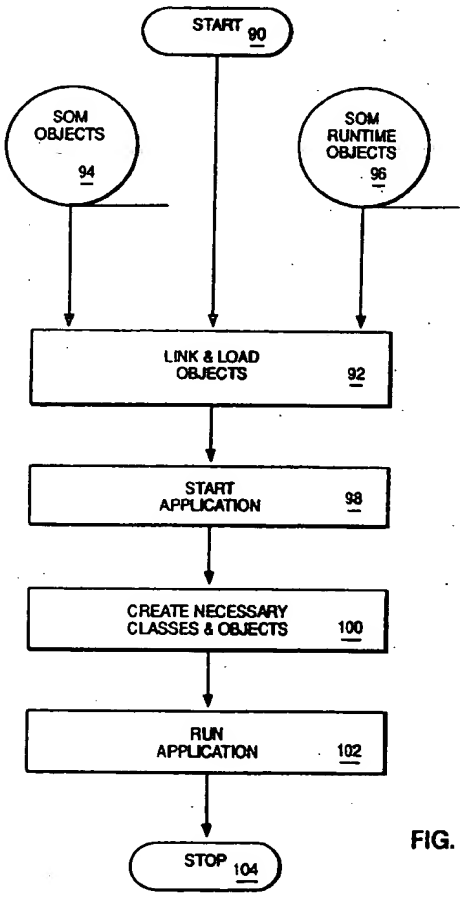

FIG. 9

EP 0 777 178 A1

Description

The present invention relates to data processing systems, and more particularly, to providing dynamic objects for object-oriented programming applications in a System Object Model (SOM).

After years of debate and discussion, object-oriented programming languages (or OOP) are becoming mainstream programming technologies. The OOP languages offer improved programmer productivity, reusable code, and easier maintenance. A central aspect of object-oriented programming models is called method resolution. This is the mechanism used to select a particular method given an object, the method's ID, and the arguments to the method. In some prior art object models, method resolution consists of determining an offset into an object's specific table of procedure entry points based on an analysis of the program's source code. These objects are referred to as static objects. Static objects do not allow a choice of which method to select during program execution. Static objects expect the choice of which method to execute for a particular method's ID, and a particular object to remain constant once program execution begins. While most static models allow the resolution to be preformed once, the selected method may be called repeatedly without going through resolution as long as the same object is to be used.

Another prior art object-oriented programming language offers a dynamic model which consists of using the name of the object to determine a specific method at runtime. In dynamic models, the choice of which method to select can change during the time of a program's execution. Unlike static objects, which allow a selected method for the same object to be called repeatedly without going through resolution, once resolution has occurred, dynamic models cannot perform such a procedure because the set of methods that a class overrides from it ancestor classes can change during a program's execution, thus changing the method that should result from resolution with a particular method's ID, and a particular object while the program is running.

In any case, static or dynamic objects must be capable of corresponding to different roles. For example, a person could represent a student, teacher, father, etc. In order to make the same entity have different responsibilities in different contexts, the prior art has use the object-oriented mechanism of multiple inheritance. Multiple inheritance represent objects as objects that inherits from different classes. Thus, objects are required to carry the overhead of their complexity in every context.

It is desirable to have a mechanism that can add roles to objects dynamically, depending on the context of the object while limiting the overhead requirements.

There is disclosed a method and apparatus for providing dynamic roles for objects in an object-oriented programming environment. A mechanism is provided that permits roles to be dynamically added for an object, depending on the context of the object which results in efficient programs. The IBM System Object Model (SOM) is the principal building block for providing this function. The IBM System Object Model implements three types of method resolution. The first is offset resolution which consists of determining an offset into an object specific table of procedure entry points based on an analysis of the program's source code. The second is dynamic resolution which consists of using the name of the object to determine a specific method at runtime. The third is dispatch resolution which is used to provide the functions of this invention. In dispatch resolution, method resolution is decided by the implementation of "somDispatch" methods. This invention overrides "somDispatch" methods in new classes to implement application specific dispatch mechanism. Dynamic objects are represented as lists of objects. Objects are added to the list at any time. For example, when a program is started, a person object will only have a student object. Before calling a function requiring teacher characteristics, a teacher role is added to the person object. When the function call returns, the teacher role is deleted from the person object if the role is no longer needed.

In summary, dynamic roles objects are associated with a role list object. Methods are provided to add a role to an object or delete a role. This is achieved by overriding the somDISPATCH method. The procedure searches for all objects in the role object list to find the first one having the needed method.

According to a first aspect of the invention, there is provided a method, implemented in a computer system, for creating a dynamic object in an object-oriented environment, comprising the steps of: creating a role list object having at least one dynamic object in said computer system in said object-oriented environment; and providing methods in said object-oriented environment to add a role to said dynamic object.

Preferably the method further comprises the steps of: determining that said role is required in an application program in said computer system in said object-oriented environment; searching said role list object in said computer system for said dynamic object containing said role.

According to a second aspect of the invention, there is provided an apparatus for creating a dynamic object in an object-oriented environment, comprising: means for creating a role list object having at least one dynamic object in said computer system in said object-oriented environment; and means for providing methods in said object-oriented environment to add a role to said dynamic object.

According to a third aspect of the invention there is provided a method implemented in a computer system for accessing a dynamic object in an object oriented environment, comprising: creating a list object in said object oriented environment; and creating a dynamic role object in said computer system in said object oriented environment having a plurality of roles as a sub-